By Ashutosh Bhatt

Introduction

The first thing we need to understand is that what is servomechanism?

Servomechanism means generating error output as a difference between output feedback and control input and reduces this error until it becomes zero – that means

• when the system reaches the desired state

• when the system produces the desired output

It is widely used in

1. Industrial applications – flat belt, motion control, load positioning, close loop control systems
2. Defense applications – gun barrel positioning, target tracking, and monitoring, guidance systems of missile or smart bomb
3. Automobile applications – automated guided vehicles (AGV), unmanned guided vehicles (UGV)
4. Aeronautic application – fly by wire system, unmanned air vehicle (UAV)
5. Power and voltage regulator applications – voltage regulator
6. Satellite application – satellite tracking

We can list out numerous such applications.

Let us understand one fascinating application in brief.

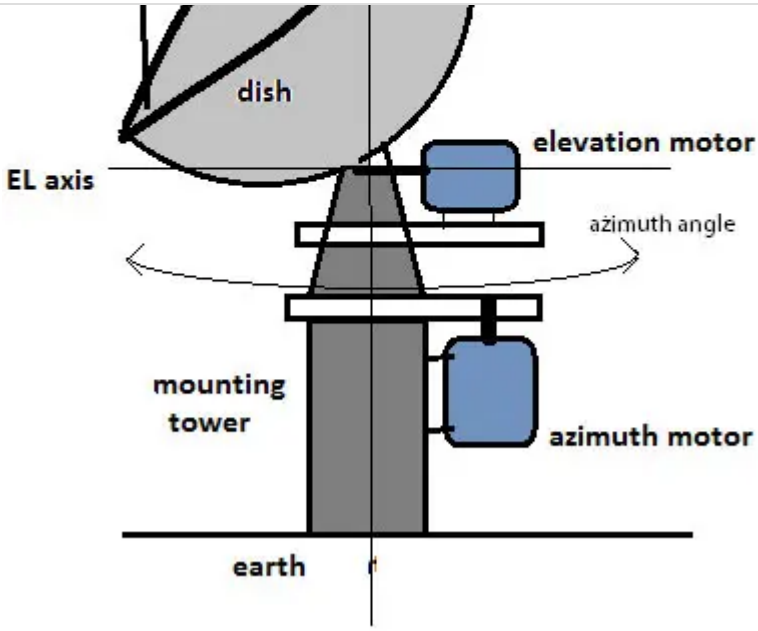
Gun barrel position system

This is one of the most required systems for the tank, anti-aircraft gun, ground base missile launcher, or howitzer. Once the target is locked in all such systems, the gun should always point to the target until the shell (bomb or missile) is fired. The target is moving (like a tank or aircraft), so according to its moving position, the gun should also automatically move. The gun movement depends on the difference generated due to the target's previous position and current position. And finally, due to accurate calculation, the bomb is fired that always hits the bull's eye.

I have discussed another such application: Automatic 2 dimensional (2D) angle (position) control for satellite dish antenna.

Automatic positioning of satellite dish antenna

The dish antenna should always point to a satellite in space to receive maximum signal. A slight deviation may also reduce signal strength. The antenna position is fixed by setting two angles (1) azimuth angle (2) elevation angle. These are set by entering their values. Once these values are entered, the dish antenna moves in the horizontal and vertical direction using 2 DC gear motors, one for each direction (angle). As it moves in either



The above figure shows a diagram of the system. The azimuth motor (AZ motor) and elevation motor (EL motor) are shown. The sensors are attached with shafts of the antenna as the dish antenna changes its position, the analog output from the sensor changes.

Once the sensor position matches the entered position in terms of azimuth angle and elevation angle, both the motors stop rotating. Afterward, if there is any deviation in the antenna position due to any circumstances, it automatically reaches the desired position using motors. If we want to change the antenna position (maybe to track other satellite), then we change azimuth angle and elevation angle, and again, the antenna moves to a new modified position.

This is the description of the actual system. It consists of big and heavy motors that can take the load, and the rotate dish antenna of size varies from 3 to 15 meters. The RVDT is chosen (or even custom made) as per the antenna's desired degree of rotation. It gives corresponding analog voltage output as per the position of antenna changes. The entire system is precisely calibrated for RVDT output as per antenna angle position in both dimensions.

For example, consider the RVDT has an angular range of -45o to +45o, and corresponding analog output is 0 to 5 V. That means for 90o change in angle, the voltage change variation is 5V. So for each degree change in angle the output voltage changes by 5 / 90 = 0.055 = 55 mV (approx.). If this RVDT is used in a system such a way that as the antenna moves from 0o to 90o, it gives the output as 0 to 5 V, then every degree change in antenna position sensed as ±55 mV change in voltage output. The control system senses this voltage and generates signals to drive motors. The motors and corresponding gear mechanism is also designed to rotate the antenna in the range from 0o to 90o (or 180o or even more) in the step of 1o in both dimensions. Generally, the elevation angle range is from 0o to 90o, and the azimuth angle range may be 0o to 180o or even completely 360o.

sensor generates one pulse per 10o of motor rotation. The figure itself describes how the entire arrangement is made, although let me discuss it step by step.

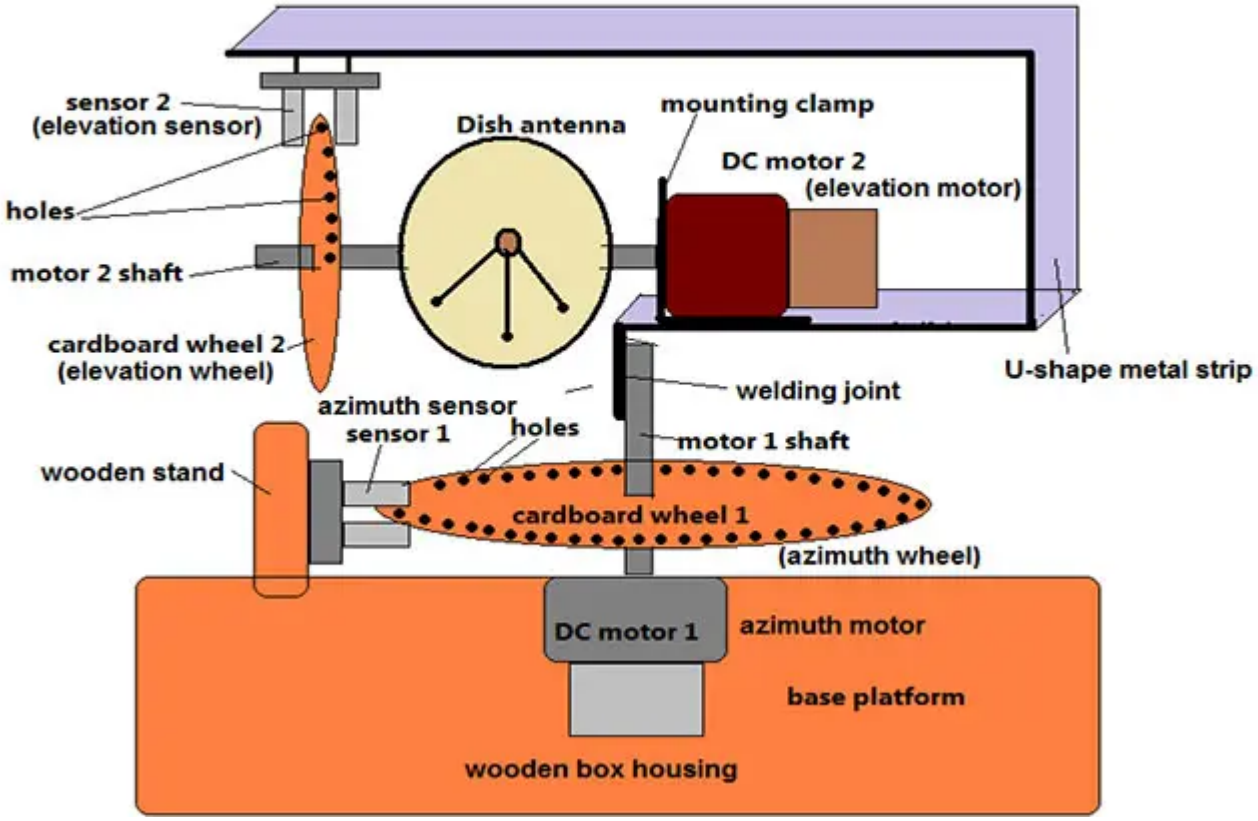
• Entire system is set up on a wooden box that is housing for the circuit and serves as a base or platform for the entire mechanism.

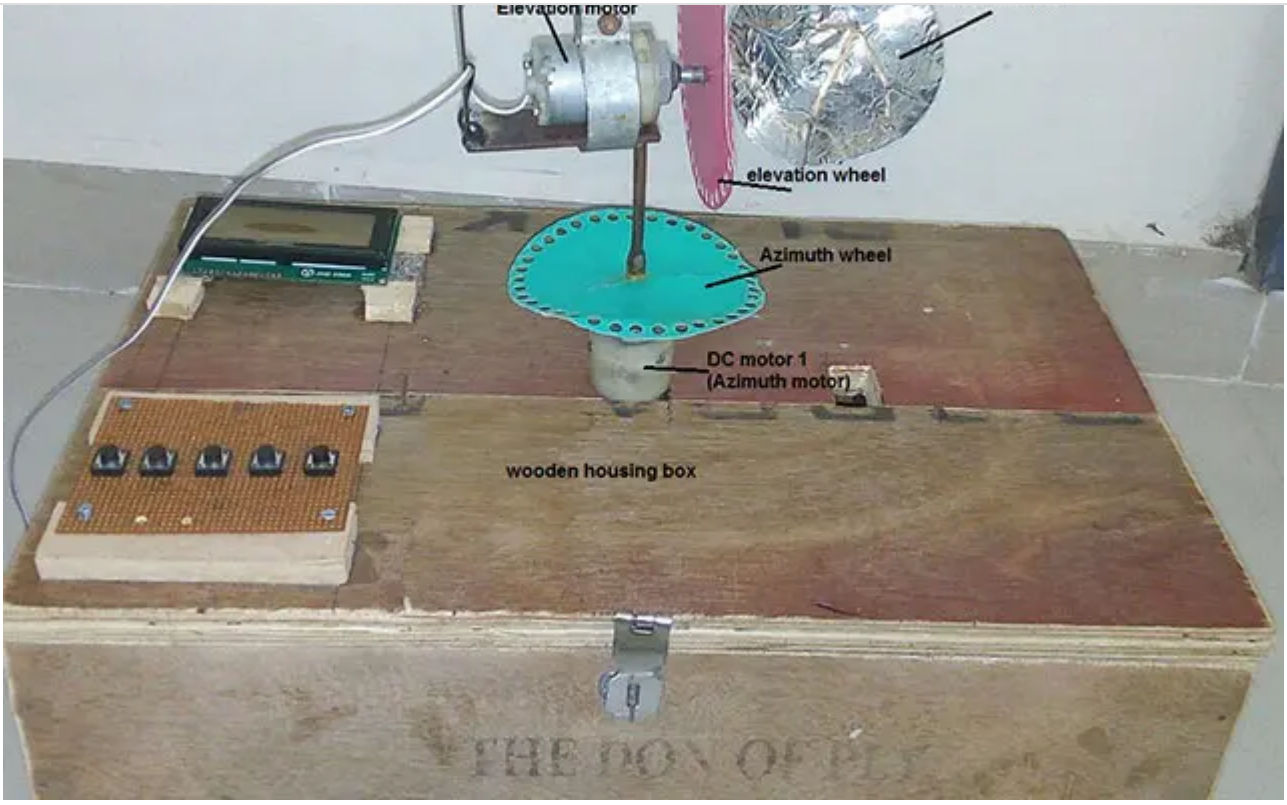
• 1st DC motor (azimuth motor) is fixed inside a wooden box using bolt and screws. The encoding wheel (azimuth wheel) is attached with its shaft as shown

• MOC7811 sensor 1 (Azimuth sensor) is also mounted vertically using a small wooden stand on the same wooden platform such a way that the holes of the azimuth encoding wheel pass through the gap of the sensor

• A U shape metal strip of around 2 inches wide is directly welded with the shaft of DC motor. one side of U shape is quite long for others, as shown.

• This strip is used to mount 2nd DC motor and another sensor. 2nd DC motor (elevation motor) is mounted on U-strip using mounting clamp and screws on the shorter side while sensor 2 (elevation sensor) is fixed at the end of the long arm (side)





Working of demonstration model: –

• Initially, the angle of both motors is set to 0o on the encoding wheel. Means dish antenna has 0o azimuth angle and 0o elevation angle

• The azimuth angle can be set from 0o to 350o in the step of 10o. The user can set the desired angle using pushbuttons

• Next, the user has to set an elevation angle. It can be set from 0o to 70o in the step of 10o using pushbuttons

• When both angles are set, the user presses another push button to rotate the dish

• As the button is pressed, the azimuth motor starts rotating to set azimuth angle. As the motor rotates, the encoding wheel also rotates, and its holes pass through the sensor gap. The sensor generates pulses that give feedback on the motor's angular position. The motor rotates till the set angle and the motor actual rotation angle becomes equal. If the set angle is more than the actual motor angle, then the motor rotates CW direction, and if the set angle is less, the motor rotates CCW direction

• Next elevation motor starts rotating. It rotates dish as well as 2nd encoding wheel. Again motor rotates till the set angle and the motor actual rotation angle becomes equal. It also rotates CW or CCW as per the set angle is more or less than the actual angle

• Thus, the dish antenna attains the desire angular position in azimuth and elevation planes

Hardware

• PORTC drives LCD data pins D0 – D7. Two control pins Rs and En or LCD are connected with PORTB pins PB4 and PB5, respectively. RW pin of LCD is connected to ground to make it write enable
• PORTB pins PB0 – PB3 drives 2 DC motors through the L293D chip. These pins are connected to the input of L293D, and two motors are connected with outputs of L293D
• A 16 MHz crystal is connected to crystal input pins of ATMega16. Two 22pf capacitors are connected for biasing and stability of the crystal
• Reset button connected as shown with reset input pin to provide a manual reset to the microcontroller
Operation: –
• Initially LCD shows set AZ angle and set EL angle as 0o. It also shows the current AZ angle and EL angle as 0o.
• User has to set the AZ angle and EL angle by pressing AZ angle inc/dec buttons and EL angle inc/dec buttons
• Whenever a button is pressed, the angle is changed by 10o
• After setting angles to rotate, the dish button is pressed. Both motors rotate one by one till the dish antenna angles match with set azimuth angle and elevation angle
Software
• Software is written in C language
• Compiled using AVR studio 4
• Tested using AVR simulatro1 for ATmega16 device (available with AVR studio 4)

Project Source Code

```
###


  //Program to

  Here is the complete program with necessary comments.

  #include

  #include

  #include

  #include

  unsigned int set_az_angle=0, set_el_angle=0,dish_az_angle=0,dish_el_angle=0, rotate_reverse_f
  lag=0,up_down_flag=0;
```

```
s++;
}
}
//////////////  function to initialize LCD  ////////////////////////////////////////
void lcd_init()
  {
    lcd_sendcmd(0x3E); // 8 bits per char
lcd_sendcmd(0x0E); // screen on and cursor ON
lcd_sendcmd(0x01); // clear LCD memory and home cursor
lcd_sendcmd(0x84); // go to column 4 of line 1
lcd_printstr("Dish antenna"); // display message
lcd_sendcmd(0xC4); // go to column 4 of line 2
lcd_printstr("angle control");
  }
//////////////////  function to display angle value on LCD ////////////////////////////
void display_value(unsigned int angle)
  {
   unsigned int t1,a;
unsigned char asci[3];
unsigned int t;
t = angle;
```

```
      asci[a]=t1+0x30;
      t=t/10;
      a--;
     }
     asci[0]=t+0x30;
      }
    else
      {
     t1=t%10;
               asci[2]=t1+0x30;
               t=t/10;
               asci[1]=t+0x30;
               asci[0]=0x20;
             }
       lcd_senddata(asci[0]);
       lcd_senddata(asci[1]);
       lcd_senddata(asci[2]);
       lcd_senddata(0xDF);
    }
////////////////////// function to increment azimuth angle value ////////////////////
void inc_az_angle()
```

```
  }
////////////////////// function to decrement azimuth angle value ////////////////////
void dec_az_angle()
  {
      if(set_az_angle>0)
            set_az_angle-=10;   // decrement set angle by 10o till its not 0o
      lcd_sendcmd(0x8D);
      display_value(set_az_angle);    // display it on LCD
  }
////////////////////// function to increment elevation angle value /////////////////
void inc_el_angle()
  {
     if(set_el_angle<70) set_el_angle+=10;
     lcd_sendcmd(0xCD);
     display_value(set_el_angle);
  }
////////////////////// function to decrement elevation angle value /////////////////
void dec_el_angle()
  {
      if(set_el_angle>0) set_el_angle-=10;
      lcd_sendcmd(0xCD);
      display_value(set_el_angle);
```

```
 else // otherwise
       rotate_reverse_flag=1; // rotate dish CCW
 if(set_el_angle>dish_el_angle)                  // if new set el angle is larger
                 up_down_flag=0;                 // move dish up
 else  // otherwise
         up_down_flag=1;  // move dish down
 if(rotate_reverse_flag==0)
    {
       //////// rotate CW dish till set az angle and dish az angle are not same
       while(set_az_angle!=dish_az_angle) PORTB = 0x01;
       PORTB = 0x00;                               // stop when same
    }
 else
    {
       // rotate CCW dish till set az angle and dish az angle are not same
       while(set_az_angle!=dish_az_angle) PORTB = 0x02;
       PORTB = 0x00;                              // stop when same
    }
if(up_down_flag==0)
    {
       // move dish up till set el angle and dish el angle are not same
```

```
            while(set_el_angle!=dish_el_angle) PORTB = 0x08;
            PORTB = 0x00;                                   // stop when same
          }
    }
int main(void)
{
    DDRC=0xFF; // initializes ports as
    DDRD=0xC3; // input or output
    DDRB=0xFF;
    DDRA=0x00;
    PORTC=0x00;
    PORTB=0x00;
    MCUCR = (1<
    GICR=(1< // enable external interrupts
    sei(); // enable global interrupts
    lcd_init(); // initialize LCD
    _delay_ms(2000); // wait for 2 sec
    lcd_sendcmd(0x01); // clear LCD
    lcd_printstr("Set AZ angle:"); // display angle values
    display_value(set_az_angle);
    lcd_sendcmd(0xC0);
```

```
    lcd_printstr("Dish EL angle:");
    display_value(set_el_angle);
loop:while(PINA==0xE0);                 // loop until no switch is pressed
    switch(PINA)
      {
         case 0xE1:                     // when switch 1 is pressed
            inc_az_angle();             // increment az angle
            _delay_ms(200);             // delay for key debounce
            break;
         case 0xE2:                     // same as above
             dec_az_angle();
             _delay_ms(200);
             break;
         case 0xE4:
            inc_el_angle();
            _delay_ms(200);
            break;
         case 0xE8:
            dec_el_angle();
            _delay_ms(200);
            break;
```

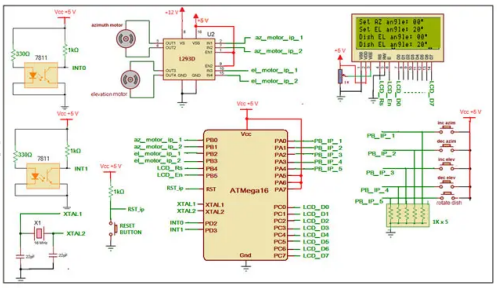
```
}
///////////////// external interrupt 0 service routine ////////////////////////////
ISR(INT0_vect)
  {
    if(rotate_reverse_flag==1) // if dish is rotating reverse
      {
        if(dish_az_angle>0) dish_az_angle-=10;       // decrement angle value by 10
      }
    else                                            // otherwise
      {
        if(dish_az_angle<360) dish_az_angle+=10;    // increment angle value by 10
      }
   lcd_sendcmd(0xA2);
   display_value(dish_az_angle);                    // display angle value on LCD
  }
///////////////// external interrupt 1 service routine ////////////////////////////
ISR(INT1_vect)
  {
    if(up_down_flag==1) // if dish is moving down
      {
         if(dish_el_angle>0) dish_el_angle-=10; // decrement angle value by 10
      }
```

```
}
```

###

Circuit Diagrams

main-circuit



Project Video